Learning Hierarchical Line Buffer for Image Processing Supplementary Document

Jiacheng Li Feiran Li Daisuke Iso
Sony Research

1. Additional Results

1.1. Additional Quantitative Results

RAW denoising. Table 1 and Table 2 present the SSIM [17] and LPIPS [22] results for raw image denoising. As shown in Table 2, our method demonstrates perceptual quality improvements consistent with the findings in terms of fidelity, further confirming its effectiveness.

Gaussian denoising. Table 3 provides results for grayscale image denoising under the Gaussian noise assumption. Our method outperforms the previous local DNN-based method, LineDL [10], highlighting its superior performance.

Super-Resolution. Table 4 and Table 5 report the PSNR and SSIM [17] results for $\times 2$ and $\times 3$ image upscaling factors, respectively.

1.2. Additional Qualitative Results

In Fig. 1 and Fig. 2, we provide more visual results on RAW and Gaussian grayscale image denoising, respectively.

2. More Implementation Details

Network architecture. Our model employs three blocks for both the intra-line encoder and decoder. The base channel dimensions for x_t, h_t, and c_t are set to 64. In each stage of the intra-line block, one-quarter of the channels are allocated for expanding strip convolutions with kernel sizes of 1×5, 1×7, and 1×9, respectively. Convolutional layers within the intra-line block are followed by LeakyReLU activation functions, and skip connections are incorporated between blocks. In the inter-line processing module, all convolutional layers utilize a 3×3 kernel. Since the public implementation of LineDL is not available, we reimplemented the method and fully trained it based on the official settings.

Training details. We train our model using the Adam optimizer [12] with a cosine annealing learning rate schedule [14]. For RAW image denoising, the model is trained for 1000 epochs with a batch size of 1 and a patch size of 512×512. For Gaussian image denoising and super-resolution tasks, training is conducted for 2×10^5 iterations with a batch size of 16 and a patch size of 128×128. For line grouping, an overlap ratio of $\frac{1}{2}$ is applied between adjacent groups, with the final result obtained by averaging the overlapping regions.

Efficiency evaluation. The shared memory demand in local methods depends on the shape, particularly the channel dimensions, of the hidden and cell states in both LineDL and our method. The memory demand reported for LineDL [10] considers only the largest feature tensor. In contrast, our evaluation utilizes the official PyTorch profiling tool (`torch.profiler.ProfilerActivity.CUDA`) and also reports the shared memory usage for hidden state features.

3. Additional Discussion

The classification of DNNs. Although certain deep neural networks (DNNs), particularly convolutional neural networks (CNNs), can operate in local regions, we categorize them as global methods due to their receptive fields (RF) often extending beyond the height of a single line group. For example, the RF of FSRCNN is 17×17, and that of UNet is about 106×106. This characteristic makes them unsuitable for line-buffer-based processing.

Running time. Since we simulate line buffers on a GPU by manually controlling data transfers with a for-loop for a comprehensive comparison with global DNNs, the advantage in latency of local DNNs cannot be enjoyed by the current implementation. The targeted streaming scenario is intrinsically suited for data fragmentation and incurs no extra runtime at the algorithm level. Image sensors naturally output data line-by-line, and our method is designed to operate as soon as sufficient lines are available, avoiding to wait for the full image to load and thus benefiting end-to-end latency. Nevertheless, we give examples of the running time here. Specifically, for denoising a 1920×1080 HD image, the running time of DnNN [20] is $101.78 \mu s$, while the baseline, LineDL, and our method takes $155.25 \mu s$, $558.90 \mu s$, and $183.60 \mu s$, respectively, under the assumption of non-overlap line groups with $l = 8$. For upsampling a 480×320 image to HD resolution (1920×1080), the running time of a typical lightweight DNN, CARN [1] is $24.05 \mu s$, while the baseline, LineDL, and our method takes $22.88 \mu s$, $88.80 \mu s$, and $40.80 \mu s$, respectively.

Type	Method	ELD		SID			Mem. Req.	
		$\times$100	$\times$200	$\times$100	$\times$250	$\times$300	Param.	Peak$_{\text{UHD}}$
Global (Classic)	NLM [3]	0.8420	0.7496	0.8477	0.7045	0.6004	-	-
	BM3D [7]	0.8463	0.7552	0.8623	0.7285	0.6287	-	-
Global (DNNs)	DnCNN [20]	0.7302	0.6387	0.5794	0.4540	0.3899	2.24MB	3.96GB
	UNet [4]	0.9700	0.9197	0.9487	0.9345	0.9198	31.04MB	1.39GB
	DRUNet [21]	0.9737	0.9520	0.9515	0.9305	0.9109	130.57MB	1.98GB
	NAFNet [5]	0.9768	0.9477	0.9517	0.9261	0.8994	116.63MB	13.94GB
	Restormer [19]	0.9776	0.9524	0.9524	0.9322	0.9118	104.51MB	OOM
Local (Classic)	Median($K = 3$) [15]	0.8073	0.6933	0.7338	0.5823	0.4820	-	0.50MB
	Median($K = 5$) [15]	0.8332	0.7311	0.7917	0.6259	0.5240	-	0.54MB
	Median($K = 7$) [15]	0.8366	0.7419	0.8204	0.6525	0.5499	-	0.57MB
	Wiener($K = 5$) [2]	0.8460	0.7501	0.8195	0.6645	0.5592	-	1.06MB
	Wiener($K = 7$) [2]	0.8468	0.7581	0.8432	0.6907	0.5827	-	1.11MB
	Bilateral($K = 5$) [16]	0.8423	0.7436	0.8108	0.6577	0.5557	-	15.06MB
	Bilateral($K = 7$) [16]	0.8470	0.7542	0.8393	0.6886	0.5848	-	29.15B
Local (DNNs)	Baseline	0.9272	0.8689	0.9257	0.8967	0.8724	6.65MB	15.0+ 0MB
	LineDL [10]	0.9595	0.9087	0.9444	0.9264	0.9092	24.36MB	15.0+120.0MB
	Ours	**0.9624**	**0.9179**	**0.9453**	**0.9291**	**0.9120**	3.37MB	8.0+ 15.0MB

Table 1. Quantitative results (SSIM) for RAW image denoising on the ELD dataset [18] and SID dataset [4]. The memory requirements are estimated assuming FP32 precsion. Peak memory is reported in UHD (3840 $\times$ 2160) resolution, and the activation peak memory usage of local DNNs is reported in the format of "current pass + shared". Local methods operate on a line group of 8, *i.e.*, $l = 8$. "OOM" denotes that out-of-memory error occurs on an NVIDIA H100 GPU with 80GB memory. The best results in local methods are **highlighted**.

Type	Method	ELD		SID			Mem. Req.	
		$\times$100	$\times$200	$\times$100	$\times$250	$\times$300	Param.	Peak$_{\text{UHD}}$
Global (Classic)	NLM [3]	0.1783	0.3462	0.1939	0.2812	0.3221	-	-
	BM3D [7]	0.0616	0.1283	0.1108	0.1763	0.2219	-	-
Global (DNNs)	DnCNN [20]	0.2356	0.3810	0.3801	0.5171	0.5499	2.24MB	3.96GB
	UNet [4]	0.0350	0.0593	0.0730	0.1006	0.1250	31.04MB	1.39GB
	DRUNet [21]	0.0322	0.0566	0.0705	0.1069	0.1363	130.57MB	1.98GB
	NAFNet [5]	0.0347	0.0592	0.0666	0.0970	0.1252	116.63MB	13.94GB
	Restormer [19]	0.0289	0.0484	0.0631	0.0955	0.1220	104.51MB	OOM
Local (Classic)	Median($K = 3$) [15]	0.2035	0.3382	0.2947	04436	0.4696	-	0.50MB
	Median($K = 5$) [15]	0.1508	0.2688	0.2494	0.3769	0.4147	-	0.54MB
	Median($K = 7$) [15]	0.1309	0.2124	0.2308	0.3425	0.3902	-	0.57MB
	Wiener($K = 5$) [2]	0.1361	0.2408	0.2351	0.3375	0.3810	-	1.06MB
	Wiener($K = 7$) [2]	0.1270	0.1895	0.2219	0.3112	0.3629	-	1.11MB
	Bilateral($K = 5$) [16]	0.1579	0.3050	0.2359	0.3559	0.3962	-	15.06MB
	Bilateral($K = 7$) [16]	0.1339	0.2441	0.2082	0.3061	0.3519	-	29.15B
Local (DNNs)	Baseline	0.0530	0.0892	0.1203	0.1823	0.2040	6.65MB	15.0+ 0MB
	LineDL [10]	0.0422	0.0693	0.0776	0.1153	0.1448	24.36MB	15.0+120.0MB
	Ours	**0.0381**	**0.0647**	**0.0750**	**0.1115**	**0.1395**	3.37MB	8.0+ 15.0MB

Table 2. Quantitative results (LPIPS) for RAW image denoising on the ELD dataset [18] and SID dataset [4]. Lower is better.

Type	Method	Set12			BSD68			Urban100			Mem. Req.	
		15	25	50	15	25	50	15	25	50	Param.	Peak_{UHD}
Global (Classic)	NLM [3]	31.19	28.50	24.79	30.06	27.49	24.47	30.79	27.64	23.41	-	-
	BM3D [7]	32.39	29.99	26.75	31.15	28.63	25.70	32.36	29.74	26.07	-	-
Global (DNNs)	DnCNN [20]	32.86	30.44	27.18	31.73	29.23	26.23	32.64	29.95	26.26	2.24MB	3.96GB
	DRUNet [4]	33.25	30.94	27.90	31.91	29.48	26.59	33.44	31.11	27.96	130.57MB	1.98GB
	SwinIR [13]	33.36	31.01	27.91	31.97	29.50	26.58	33.70	31.30	27.98	47.58MB	50.31GB
	Restormer [19]	33.42	31.08	28.00	31.96	29.52	26.62	33.79	31.46	28.29	104.51MB	OOM
	HAT [6]	33.49	31.13	28.07	31.99	29.52	26.60	33.99	31.67	28.62	83.09MB	OOM
Local (Classic)	Median($K = 3$) [15]	27.31	25.19	21.02	26.87	24.89	20.94	25.24	23.62	20.24	-	0.50MB
	Wiener($K = 5$) [2]	27.78	26.44	23.29	27.34	26.12	23.13	25.89	24.50	21.92	-	1.06MB
	Bilateral($K = 5$) [16]	28.61	26.58	23.89	27.74	26.14	23.67	26.26	24.31	22.05	-	15.06MB
Local (DNNs)	Baseline	32.24	29.62	26.10	31.28	28.65	25.51	31.94	28.99	25.00	6.65MB	15.0+ 0MB
	LineDL [10]	32.46	29.97	26.72	31.42	28.90	25.90	32.10	29.36	**25.78**	24.36MB	15.0+120.0MB
	Ours	**32.70**	**30.21**	**26.87**	**31.62**	**29.08**	**26.04**	**32.38**	**29.46**	25.73	3.37MB	8.0+ 15.0MB

Table 3. Quantitative results (PSNR) for Gaussian grayscale image denoising on standard benchmark datasets.

Type	Method	Set5		Set14		BSDS100		Urban100		Manga109		Mem. Req.	
		PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	Param.	Peak_{HD}
Global (DNNs)	FSRCNN [8]	37.05	0.9560	32.66	0.9090	31.53	0.8902	29.88	0.9020	36.67	0.9710	0.05MB	442.97MB
	CARN [1]	37.76	0.9590	33.52	0.9166	32.09	0.8978	31.92	0.9256	38.36	0.9765	4.45MB	506.25MB
	SwinIR [13]	38.14	0.9611	33.86	0.9206	32.31	0.9012	32.76	0.9340	39.12	0.9783	47.58MB	50.31GB
	HAT [6]	38.73	0.9637	35.13	0.9282	32.69	0.9060	34.81	0.9489	40.71	0.9819	83.09MB	OOM
	MambIR [9]	38.57	0.9627	34.67	0.9261	32.58	0.9048	34.15	0.9446	40.28	0.9806	18.61MB	64.00GB
Local (Classic)	Nearest	30.82	0.8991	28.51	0.8446	28.39	0.8239	25.62	0.8199	28.12	0.9089	-	0.96MB
	Bilinear	32.12	0.9106	29.15	0.8384	28.65	0.8090	25.95	0.8077	29.13	0.9115	-	0.96MB
	Bicubic [11]	33.63	0.9292	30.23	0.8681	29.53	0.8421	26.86	0.8394	30.78	0.9338	-	0.96MB
Local (DNNs)	Baseline	37.32	0.9573	33.03	0.9118	31.68	0.8927	30.87	0.9148	37.03	0.9727	6.67MB	8.5+ 0MB
	LineDL [10]	37.59	0.9586	33.19	0.9138	31.84	0.8951	31.22	0.9189	37.44	0.9740	27.45MB	75.0+60.0MB
	Ours	**37.63**	**0.9589**	**33.32**	**0.9149**	**31.90**	**0.8960**	**31.43**	**0.9213**	**37.58**	**0.9746**	3.39MB	3.8+ 7.5MB

Table 4. Quantitative results (PSNR and SSIM) for super-resolution ($\times 2$) on the standard benchmark datasets. Peak memory is reported in upscaling HD (1920×1080) resolution with a factor of 4, assuming FP32 precsion.

Type	Method	Set5		Set14		BSDS100		Urban100		Manga109		Mem. Req.	
		PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	Param.	Peak_{HD}
Global (DNNs)	FSRCNN [8]	33.18	0.9140	29.37	0.8240	28.53	0.7910	26.43	0.8080	31.10	0.9210	0.05MB	442.97MB
	CARN [1]	34.29	0.9255	30.29	0.8407	29.06	0.8034	28.06	0.8493	33.50	0.9440	4.45MB	506.25MB
	SwinIR [13]	34.62	0.9289	30.54	0.8463	29.20	0.8082	28.66	0.8624	33.98	0.9478	47.58MB	50.31GB
	HAT [6]	35.16	0.9335	31.33	0.8576	29.59	0.8177	30.70	0.8949	35.84	0.9567	83.09MB	OOM
	MambIR [9]	35.08	0.9323	30.99	0.8536	29.51	0.8157	29.93	0.8841	35.43	0.9546	18.61MB	64.00GB
Local (Classic)	Nearest	27.93	0.8123	26.00	0.7330	26.17	0.7065	23.34	0.6992	25.04	0.8157	-	0.96MB
	Bilinear	29.54	0.8504	26.96	0.7526	26.77	0.7177	23.99	0.7135	26.15	0.8372	-	0.96MB
	Bicubic [11]	30.40	0.8678	27.55	0.7736	27.20	0.7379	24.45	0.7343	26.94	0.8554	-	0.96MB
Local (DNNs)	Baseline	33.64	0.9191	29.82	0.8321	28.68	0.7947	27.22	0.8308	32.16	0.9324	6.67MB	8.5+ 0MB
	LineDL [10]	33.84	0.9217	29.99	0.8353	28.82	0.7981	27.46	0.8362	32.48	0.9359	27.45MB	75.0+60.0MB
	Ours	**33.92**	**0.9224**	**30.04**	**0.8367**	**28.85**	**0.7994**	**27.62**	**0.8410**	**32.69**	**0.9379**	3.39MB	3.8+ 7.5MB

Table 5. Quantitative results (PSNR and SSIM) for super-resolution ($\times 3$) on the standard benchmark datasets. Peak memory is reported in upscaling HD (1920×1080) resolution with a factor of 4, assuming FP32 precsion.

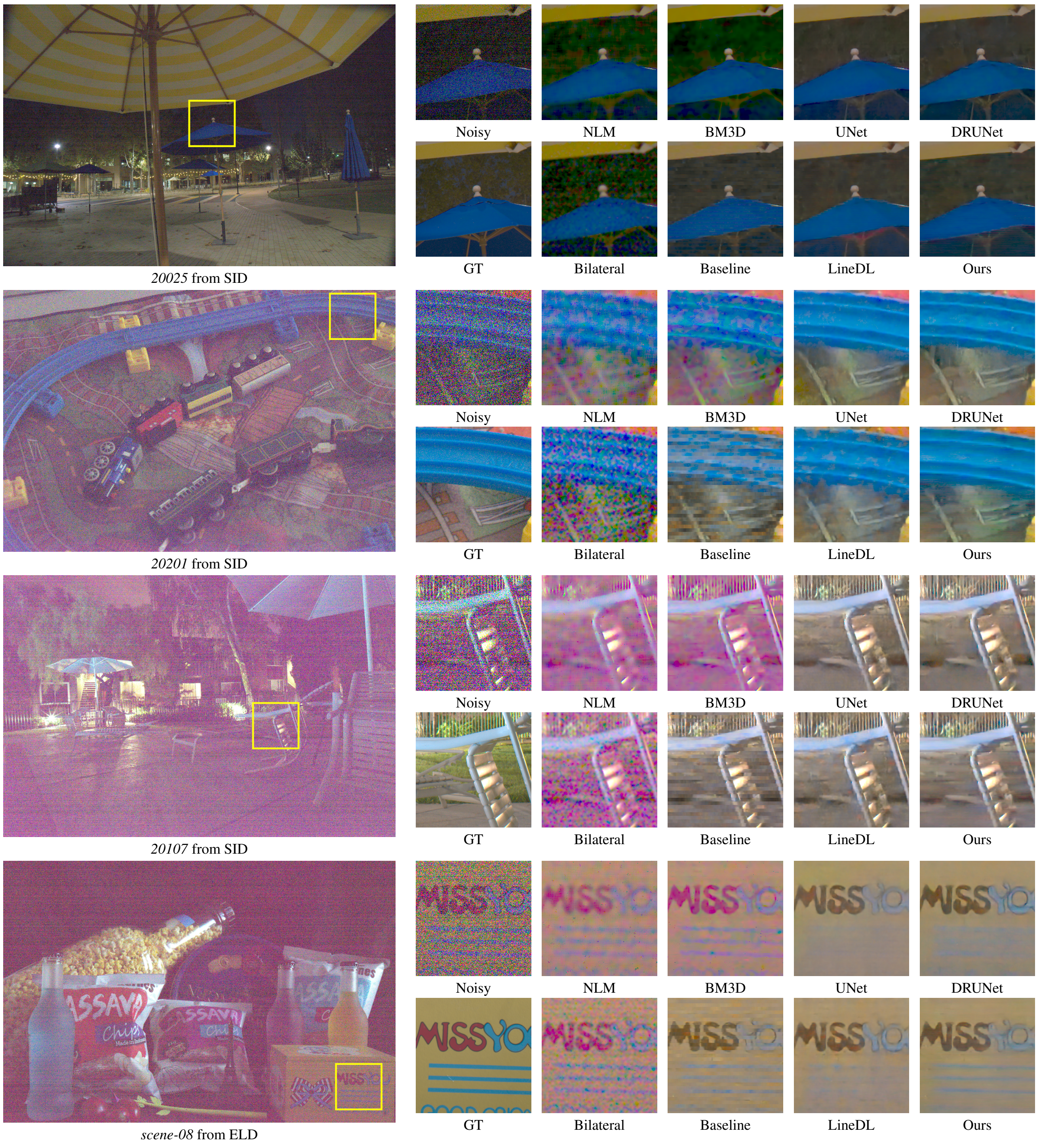

Figure 1. Additional qualitative results for RAW image denoising.

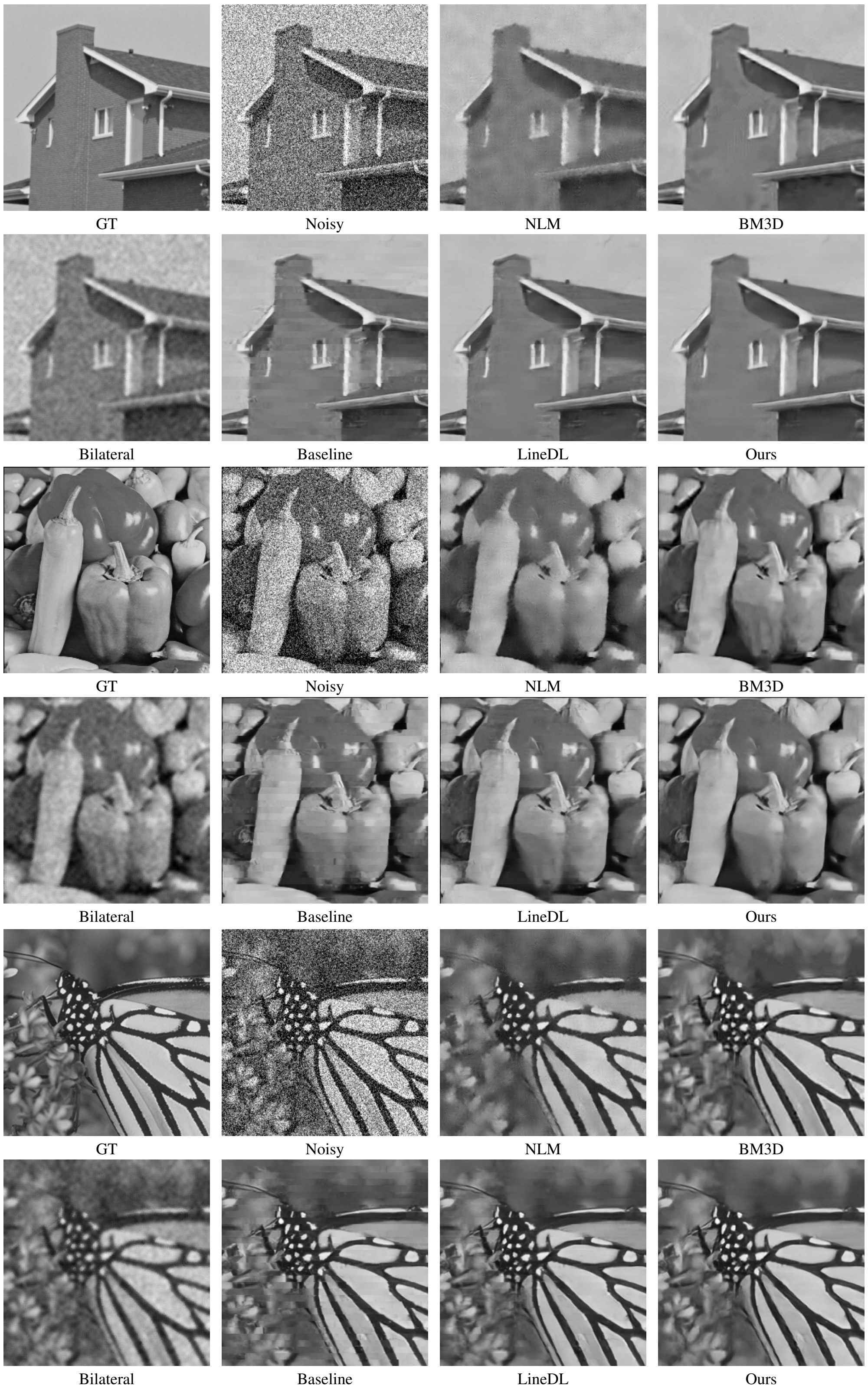

Figure 2. Qualitative results for Gaussian image denoising.

References

[1] Namhyuk Ahn, Byungkon Kang, and Kyung-Ah Sohn. Fast, accurate, and lightweight super-resolution with cascading residual network. In *ECCV*, 2018. 1, 4

[2] Jacob Benesty, Jingdong Chen, and Yiteng Huang. Study of the widely linear wiener filter for noise reduction. In *ICASSP*, 2010. 2, 3

[3] Antoni Buades, Bartomeu Coll, and Jean-Michel Morel. A non-local algorithm for image denoising. In *CVPR*, 2005. 2, 3

[4] Chen Chen, Qifeng Chen, Jia Xu, and Vladlen Koltun. Learning to see in the dark. In *CVPR*, 2018. 2, 3

[5] Liangyu Chen, Xiaojie Chu, Xiangyu Zhang, and Jian Sun. Simple baselines for image restoration. In *ECCV*, 2022. 2

[6] Xiangyu Chen, Xintao Wang, Jiantao Zhou, Yu Qiao, and Chao Dong. Activating more pixels in image super-resolution transformer. In *CVPR*, 2023. 3, 4

[7] Kostadin Dabov, Alessandro Foi, Vladimir Katkovnik, and Karen O. Egiazarian. Image denoising by sparse 3-d transform-domain collaborative filtering. *IEEE Trans. Image Process.*, 16(8):2080–2095, 2007. 2, 3

[8] Chao Dong, Chen Change Loy, and Xiaoou Tang. Accelerating the super-resolution convolutional neural network. In *ECCV*, 2016. 4

[9] Hang Guo, Jinmin Li, Tao Dai, Zhihao Ouyang, Xudong Ren, and Shu-Tao Xia. Mambair: A simple baseline for image restoration with state-space model. In *ECCV*, 2024. 4

[10] Yujie Huang, Wenshu Chen, Liyuan Peng, Yuhao Liu, Mingyu Wang, Xiao-Ping (Steven) Zhang, and Xiaoyang Zeng. Linedl: Processing images line-by-line with deep learning. *IEEE Trans. Image Process.*, 32:3150–3162, 2023. 1, 2, 3, 4

[11] R. Keys. Cubic convolution interpolation for digital image processing. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 29:1153–1160, 1981. 4

[12] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *ICLR*, 2015. 1

[13] Jingyun Liang, Jiezhang Cao, Guolei Sun, Kai Zhang, Luc Van Gool, and Radu Timofte. Swinir: Image restoration using swin transformer. In *ICCV Workshops*, 2021. 3, 4

[14] Ilya Loshchilov and Frank Hutter. SGDR: stochastic gradient descent with warm restarts. In *ICLR*, 2017. 1

[15] Ioannis Pitas and Anastasios N Venetsanopoulos. *Nonlinear digital filters: principles and applications*. Springer Science & Business Media, 2013. 2, 3

[16] Carlo Tomasi and Roberto Manduchi. Bilateral filtering for gray and color images. In *ICCV*, 1998. 2, 3

[17] Zhou Wang, Alan C. Bovik, Hamid R. Sheikh, and Eero P. Simoncelli. Image quality assessment: from error visibility to structural similarity. *IEEE Trans. Image Process.*, 13(4): 600–612, 2004. 1

[18] Kaixuan Wei, Ying Fu, Yinqiang Zheng, and Jiaolong Yang. Physics-based noise modeling for extreme low-light photography. *IEEE Trans. Pattern Anal. Mach. Intell.*, 44(11): 8520–8537, 2022. 2

[19] Syed Waqas Zamir, Aditya Arora, Salman Khan, Munawar Hayat, Fahad Shahbaz Khan, and Ming-Hsuan Yang. Restormer: Efficient transformer for high-resolution image restoration. In *CVPR*, 2022. 2, 3

[20] Kai Zhang, Wangmeng Zuo, Yunjin Chen, Deyu Meng, and Lei Zhang. Beyond a gaussian denoiser: Residual learning of deep CNN for image denoising. *IEEE Trans. Image Process.*, 26(7):3142–3155, 2017. 1, 2, 3

[21] Kai Zhang, Yawei Li, Wangmeng Zuo, Lei Zhang, Luc Van Gool, and Radu Timofte. Plug-and-play image restoration with deep denoiser prior. *IEEE Trans. Pattern Anal. Mach. Intell.*, 44(10):6360–6376, 2022. 2

[22] Richard Zhang, Phillip Isola, Alexei A. Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *CVPR*, 2018. 1